

Fixing improper colorings of graphs*

Valentin Garnero¹, Konstanty Junosza-Szaniawski², Mathieu Liedloff¹, Pedro Montealegre¹, and Paweł Rzążewski^{†2,3}

¹*Université d'Orléans, INSA Centre Val de Loire, LIFO, 45067 Orléans, France., E-mail: {valentin.garnero, mathieu.liedloff, pedro.montealegre}@univ-orleans.fr*

²*Warsaw University of Technology, Faculty of Mathematics and Information Science, Koszykowa 75, 00-662 Warszawa, Poland., E-mail: {k.szaniawski, p.rzazewski}@mini.pw.edu.pl*

³*Institute of Computer Science and Control, Hungarian Academy of Sciences (MTA SZTAKI), Budapest, Hungary*

July 26, 2016

Abstract

In this paper we consider a variation of a recoloring problem, called the Color-Fixing. Let us have some non-proper r -coloring φ of a graph G . We investigate the problem of finding a proper r -coloring of G , which is “the most similar” to φ , i.e. the number k of vertices that have to be recolored is minimum possible. We observe that the problem is NP-complete for any $r \geq 3$, even for bipartite planar graphs. On the other hand, the problem is fixed-parameter tractable, when parameterized by the number of allowed transformations k . We provide an $2^n \cdot n^{O(1)}$ algorithm for the problem (for any fixed r) and a linear algorithm for graphs with bounded treewidth. We also show several lower complexity bounds, using standard complexity assumptions. Finally, we investigate the *fixing number* of a graph G . It is the maximum possible distance (in the number of transformations) between some non-proper coloring of G and a proper one.

*An extended abstract of this paper was presented on the conference SOFSEM 2015 [22].

[†]Supported by ERC Starting Grant PARAMTIGHT (No. 280152).

1 Introduction

Many problems in real-life applications have a dynamic nature. When the constraints change, the previously found solution may no longer be optimal or even feasible. Therefore often there is needed to recompute the solution (preferably using the old one). This variant is called a *reoptimization* and has been studied for many combinatorial problems, e.g. TSP (see Ausiello *et al.* [1]), Shortest Common Superstring (see Bilò *et al.* [2]) or Minimum Steiner Tree (see Zych and Bilò [28]). We also refer the reader to the paper of Shachnai *et al.* [27], where the authors describe a general model for combinatorial reoptimization.

Another family of problems, in which we deal with transforming one solution to another, is *reconfiguration*. Here we are given two feasible solutions and want to transform one into another by a series of simple transformations in such a way that every intermediate solution is feasible (see e.g. Ito *et al.* [20]). When we consider a reconfiguration version of the graph coloring problem, we want to transform one proper coloring into another one in such a way that at every step we can recolor just one vertex and the coloring obtained after this change is still proper.

A special attention has been paid to determining if a given graph G is *r-mixing*, i.e., if for any two proper r -colorings of G you can transform one into another (maintaining a proper r -coloring at each step). Cereceda *et al.* [10, 11, 12] characterize graphs, which are 3-mixing and they provide a polynomial algorithm for recognizing them. Determining if a graph is r -mixing is PSPACE-complete for every $r \geq 4$ [8]. There are also some results showing that a graph G is $f(G)$ -mixing, where $f(G)$ is some invariant of G . For example, Jerrum [21] showed that every graph G is $(\Delta(G) + 2)$ -mixing. This bound was later refined by Bonamy and Bousquet [7], who proved that every graph is $(\chi_g(G) + 1)$ -mixing, where $\chi_g(G)$ denotes the Grundy number of G , i.e., the highest possible number of colors used by a greedy coloring of G . Clearly $\chi_g(G) \leq \Delta(G) + 1$.

Another direction of research in r -mixing graphs is the maximum number of transformations necessary to obtain one r -coloring from another one (i.e., the distance between those colorings). Bonamy and Bousquet [7] show that if $r \geq \text{tw}(G) + 2$ (where $\text{tw}(G)$ denotes the *treewidth* of G), then any two r -colorings of G are in distance of at most $2(n^2 + n)$, while for $r \geq \chi_g(G) + 1$, any two r -colorings are in distance of at most $4 \cdot \chi_g(G) \cdot n$.

A slightly different problem has been considered by Felsner *et al.* [16]. They also transformed one r -coloring to another one using some local changes, but did not require the initial coloring to be proper (the final one still has

to be proper). Also, a vertex could be recolored to color x if it did not have any neighbor colored with x (strictly speaking, any out-neighbor, as the authors were considering directed graphs). They showed that if G is a 2-orientation (i.e., every out-degree is equal to 2) of some maximal bipartite planar graph (i.e., a plane quadrangulation), then every proper 3-coloring of G could be reached in $\mathcal{O}(n^2)$ steps from any initial (even non-proper) 3-coloring of G . Similar results hold for 4-colorings and 3-orientations of maximal planar graphs (i.e., triangulations).

In this paper we consider a slightly different problem. We start with some (possibly non-proper) r -coloring and ask for the minimum number of transformations needed to obtain a proper r -coloring (any proper r -coloring, not the specific one). We are allowed to change colors of vertices arbitrarily, provided that we recolor just one vertex in each step. We mainly focus on the computational aspects of determining if, starting with some given r -coloring of G , we can reach a proper r -coloring in at most k steps.

The paper is organized as follows. In Section 3 we show that our problem is NP-complete for any $r \geq 3$, even if the input graph is planar and bipartite (here k is a part of the input). In Section 3.2 we provide an $2^n \cdot n^{\mathcal{O}(1)}$ algorithm for the problem and show that it is essentially optimal under the ETH. In the next two Sections we focus on the parameterized complexity (we refer the reader to [15, 13] for an introduction to the parameterized complexity theory). First, we show that our problem is FPT, when parameterized by $k + r$ (Section 4). We also show that for any $r \geq 3$, the r -FIX problem does not admit a polynomial kernel (unless $NP \subseteq coNP/poly$), even in the input graph is bipartite. Moreover, we provide an algorithm solving the problem for graphs with bounded treewidth (Section 4.2) and show that it is essentially optimal, under the SETH. In Tables 1 and 2 you can find a summary of the most important results of the paper.

The last section of the paper, Section 5, is purely combinatorial. We investigate the *fixing number* of G , i.e., the maximum (over all initial colorings φ) distance from φ to a proper coloring of G . We provide some combinatorial bounds and suggest directions for future research.

$r \leq 2$	$r \geq 3$	$k + r$	tw + r
P (Prop 4)	NP-c (Th 8)	FPT (Th 15)	FPT (Th 19)

Table 1: The summary of parameterized complexity results for the r -FIX problem and different parameters.

	n	$k + r$	$\text{tw} + r$
upper bound	2^n (Cor 10)	$2^{O(k \log r)}$ (Th 14)	r^t (Th 19)
lower bound	$2^{o(n)}$ (Cor 11)	$2^{o(r+k)}$ (Cor 16)	$(r - \epsilon)^t$ (Cor 21)

Table 2: The upper and lower complexity bounds for the problem under different parameterizations. We suppress polynomial factors. Lower bounds should be read: “there is no algorithm with this complexity, unless the ETH/SETH fail”.

2 Preliminaries

For a natural number r , by $[r]$ we denote the set $\{1, 2, \dots, r\}$. By an r -coloring of a graph G we mean any assignment of natural numbers from $[r]$ (called *colors*) to vertices of G . A coloring is *proper* if no two adjacent vertices get the same color. Note that there may be some colors that are not assigned to any vertex.

2.1 Considered problems

For two r -colorings φ, φ' , let $\varphi \ominus \varphi'$ denote the set $\{v \in V : \varphi(v) \neq \varphi'(v)\}$. We also define the *distance* $\text{dist}(\varphi, \varphi')$ between two r -colorings φ, φ' . It is equal to their Hamming distance, i.e. $|\varphi \ominus \varphi'|$.

The problem we consider in this paper is formally defined as follows.

Problem: Color-Fixing (FIX)

Instance: A graph G , integer k , integer r , an r -coloring φ of $V(G)$.

Question: Does there exist a proper r -coloring φ' of G such that $\text{dist}(\varphi, \varphi') \leq k$?

If r is a fixed integer, we have the following version of the problem.

Problem: r -Color-Fixing (r -FIX)

Instance: A graph G , integer k , an r -coloring φ of $V(G)$.

Question: Does there exist a proper r -coloring φ' of G such that $\text{dist}(\varphi, \varphi') \leq k$?

Such a coloring φ' is called a *witness* of an instance \mathcal{I} of FIX (r -FIX, resp.). Obviously, if $r < \chi(G)$ (by $\chi(G)$ we denote the *chromatic number* of G , i.e., the smallest number of colors needed to color G properly), then the answer is always NO. By a *recoloring* of a vertex v we mean an operation of changing the color assigned to v , obtaining another coloring φ' , such that

$$\varphi \ominus \varphi' = \{v\}.$$

In the optimization version of the problem we ask for the minimum number k of recolorings needed to transform φ into a proper coloring of G . Let $\overline{\chi}_\varphi^r(G)$ denote this minimum possible value of k . If $r < \chi(G)$, we define $\overline{\chi}_\varphi^r(G) := \infty$ for every r -coloring φ of G .

2.2 Conflict graph

For an improper coloring φ of G , let G^φ denote a *conflict graph* of G under the coloring φ , i.e., the subgraph of G induced by the set of edges $\{uv \in E(G) : \varphi(u) = \varphi(v)\}$. Note that the conflict graph can be found in polynomial time. The simple observation below will prove useful.

Observation 1. *To make the coloring φ of G proper, we need to recolor at least one endvertex of each edge of G^φ .*

Thus we observe that there is a close relation between fixing an improper coloring and finding a vertex cover in the conflict graph. This relation is explicitly described in the following lemma.

Lemma 2. *Let $G = (V, E)$ be a graph on n vertices and let $k \leq n$. Let φ be the coloring of G , such that $\varphi(v) = k + 1$ for all $v \in V$. Then $(G, k, k + 1, \varphi)$ is a YES-instance of the FIX problem if and only if G has a vertex cover of size at most k .*

Proof. We can assume that G has no isolated vertices, since removing them does not change the size of the minimum vertex cover. Define $r = k + 1$. Observe that $G^\varphi = G$.

First suppose that (G, k, φ) is a YES instance of FIX with a witness φ' . Let $S = \varphi \ominus \varphi'$, clearly $|S| \leq k$. By Observation 1 S contains at least one vertex from each edge from $G^\varphi = G$. Thus S is a vertex cover of G , of size at most k and thus (G, k) is a YES-instance of VERTEX COVER.

Now suppose that (G, k) is a YES-instance of VERTEX COVER and let $S = \{v_1, v_2, \dots, v_k\}$ the the vertex cover of size k (note that we can always add some vertices to a smaller vertex cover obtain a vertex cover of size exactly k). Define a coloring φ' of G in the following way:

$$\varphi'(v) = \begin{cases} i & \text{if } v = v_i \in S \\ r & \text{if } v \notin S. \end{cases}$$

Note that $|\varphi \ominus \varphi'| = |S| = k$ and φ' is a proper r -coloring of G . This shows that (G, k, r, φ) is a YES-instance of FIX. \square

2.3 Computational assumptions

When proving hardness results or lower bounds for algorithms, we often use some additional assumptions. The standard assumption for distinguishing easy and hard problem is $P \neq NP$. However, this assumption is often too weak to give us any insights into possible complexity of an algorithm solving our (NP-hard) problem.

So often researchers use stronger assumptions to investigate hard problems in more detail. Such an assumption, typically used for this purpose, is the so-called *Exponential Time Hypothesis* (usually referred to as the ETH), formulated by Impagliazzo and Paturi [18]. We refer the reader to the survey by Lokshtanov and Marx for more information about ETH and conditional lower bounds [26]. The version we present below (and is most commonly used) is not the original statement of this hypothesis, but its weaker version (see also Impagliazzo, Paturi, and Zane [19]).

Exponential Time Hypothesis (Impagliazzo and Paturi [18]). *There is no algorithm solving every instance of 3-SAT with N variables and M clauses in time $2^{o(N+M)}$.*

A stronger complexity assumption is the so-called *Strong Exponential Time Hypothesis*, also introduced by Impagliazzo and Paturi [18]. The version we present below is again the consequence of the original statement. It is worth mentioning that the SETH is indeed a stronger version of the ETH, as the SETH implies the ETH.

Strong Exponential Time Hypothesis (Impagliazzo and Paturi [18]). *For any $\epsilon > 0$, there is no algorithm solving every instance of CNF-SAT with N variables in time $(2 - \epsilon)^N \cdot N^{\mathcal{O}(1)}$.*

The last assumption we use (in the proof of Theorem 17) is $NP \not\subseteq coNP/poly$. It can be seen as a stronger variant of $NP \neq coNP$, which in turn implies $P \neq NP$. It is worth mentioning that $NP \subseteq coNP/poly$ implies the collapse of the polynomial hierarchy to its third level. This assumption is an important part of the framework used for showing a non-existence of polynomial kernels for parameterized problems, introduced by Bodlaender, Jansen, and Kratsch [5]. The version of the framework presented below is a special case of the original one, yet it is enough for our purpose.

Let Π be a graph problem, whose instance is a graph G . Let Π^* be a parameterized problem, whose instance is (G, k) , where G is a graph and k is an integer (parameter). We say that Π *cross-composes* into Π^* , if there exists an algorithm, which, given t instances G_1, G_2, \dots, G_t of Π , work in

time polynomial in $\sum_{i=1}^t |G_i|$, and produces an instance (G^*, k^*) of Π^* , such that:

- (G^*, k^*) is a YES-instance of Π^* iff there exists $i \in [t]$ such that G_i is a YES-instance of Π ,
- $k^* \leq p(\max_{i=1}^t |V(G_i)| + \log t)$, where p is some polynomial function.

Intuitively, we encode many (exactly t) instances of the problem Π into one instance of Π^* , such that the size of the parameter k^* is relatively small. Note that the size of G^* can be huge. The theorem below shows that such a cross-composition can be used to refute the existence of a polynomial kernel for Π^* .

Theorem 3 (Bodlaender, Jansen, and Kratsch [5]). *Let Π be an NP-hard problem and let Π^* be a parameterized problem, such that Π cross-composes into Π^* . If Π^* has a polynomial kernel, then $NP \subseteq coNP/poly$.*

We refer the reader to the handbook [13, Chapters 14 and 15] for more information about complexity assumptions.

3 Classical complexity results

In this section we show that the r -FIX problem is NP-complete for all $r \geq 3$, even for restricted input graphs. Furthermore, we show how to adapt the known exact algorithm for computing partitions of graphs to solve r -FIX.

3.1 Computational hardness of the problem

First, observe that for $r = 1$ and $r = 2$ the problem is easy.

Proposition 4. *The r -FIX problem is polynomially solvable for $r \leq 2$.*

Indeed, if $r = 1$, the problem clearly reduces to determining if the graph has no edges. For $r = 2$ the problem is also polynomial time solvable. If G is not bipartite, the answer is NO.

Proposition 5. *Let G be a bipartite graph with bipartition classes X and Y and let φ be a 2-coloring of G . Then we have*

$$\overline{\chi}_{\varphi}^2(G) = \sum_{\substack{C: \text{ connected} \\ \text{component of } G}} \min\{ |(X \ominus \varphi^{-1}(1)) \cap V(C)|, |(X \ominus \varphi^{-1}(2)) \cap V(C)| \}.$$

Proof. Let C be a connected component of G and let X', Y' denote its classes of bipartition. By φ' we denote the restriction of φ to C . To obtain a proper coloring of C , we either have to recolor the vertices from $X' \setminus \varphi'^{-1}(1)$ to color 1 and vertices from $Y' \setminus \varphi'^{-1}(2)$ to color 2, or the other way around. Therefore the minimum number of recoloring operations needed to obtain a proper coloring of C is equal to $\min\{|X' \ominus \varphi'^{-1}(1)|, |X' \ominus \varphi'^{-1}(2)|\}$. Clearly $X' \ominus \varphi'^{-1}(1) = (X \cap V(C)) \ominus (\varphi^{-1}(1) \cap V(C)) = (X \ominus \varphi^{-1}(1)) \cap V(C)$ (and symmetrically for $\varphi^{-1}(2)$). We repeat this for every connected component C of G . \square

Observe that if k is the part of the input, the problem r -FIX is NP-complete for any $r \geq 3$. Clearly, the problem is in NP (even if r is given in the input). Also, a graph G with n vertices is r -colorable if and only if G can be recolored from any fixed coloring within at most n steps. Thus the r -FIX problem is NP-complete for any $r \geq 3$ (when the number k of allowed recolorings is a part of the input). From this simple reduction, we obtain the following lemma, that will be useful later.

Lemma 6. *If there exists an algorithm solving the r -FIX problem on a graph G in time $f(G)$, then the r -COLORING problem on G can be solved in time $f(G) \cdot n^{\mathcal{O}(1)}$, when n is the number of vertices of G .*

To derive slightly stronger complexity results, consider the problem r -PREEXT defined as follows:

Problem: r -Precolor-Extension (r -PREEXT)
Instance: A graph G , a set $U \subseteq V(G)$ and a proper r -coloring φ_U of $G[U]$.
Question: Does there exist a proper r -coloring φ of G such that $\varphi(u) = \varphi_U(u)$ for all $u \in U$?

As shown by Kratochvíl [24], 3-PREEXT is NP-Complete, even if the input graph is planar and bipartite.

Let (G, U, φ_U) be an instance of 3-PREEXT, where G has n vertices. Let H and φ be the graph and its coloring defined as follows. First, pick a copy of G and color its vertices with color 1. For each $v \in V(G)$, add $(r - 3)$ groups of $(n + 1)$ pending vertices, where the i -th group is colored with the color $3 + i$. Finally, for each node $u \in U$, add two more groups of $(n + 1)$ pending vertices, each group colored with a different color from $\{1, 2, 3\} \setminus \{\varphi_U(u)\}$.

Lemma 7. *Fix $r \geq 3$. Let (G, S, φ_U) be an instance of 3-PREEXT, and $H = H(G)$, $\varphi = \varphi(\varphi_U)$ defined as above. Then (G, U, φ_U) is a YES-instance of 3-PREEXT if and only if (H, n, φ) is a YES-instance of r -FIX.*

Proof. First suppose that (G, U, φ_U) is a YES-instance of 3-PREEXT with a witness φ' , i.e., a φ' is a proper 3-coloring of G (with colors in $\{1, 2, 3\}$) that extends φ_U . Let φ'' be coloring of H where the vertices in the copy of G in H are colored according to φ' , and the other vertices of H are colored according to φ . Notice that $|\varphi'' \ominus \varphi| \leq n$, and φ'' is a proper coloring of H . Indeed, by definition of φ' no obstruction can exist between nodes in G , and by definition of φ no obstruction can exist between a vertex and their pending vertices, since the pending vertices are colored in $\{4, \dots, r\}$ if $v \in V(G) - U$ and $[r] - \varphi_U(v)$ if $v \in U$.

Conversely, suppose that (H, n, φ) is a YES-instance of r -FIX with a witness φ'' , i.e., φ'' is a proper r -coloring of H such that $|\varphi'' \ominus \varphi| \leq n$. Recall that in the coloring φ , each vertex of the copy of G in H has $n+1$ neighbors in color i for each $i \in \{4, \dots, r\}$. Thus φ'' must use only colors $\{1, 2, 3\}$ on those vertices. Moreover, the vertices u in the copy of U in H have $n+1$ neighbors in color i for each $i \in [r] \setminus \{\varphi_U(u)\}$. Then the restriction of φ'' to $V(G)$ is a 3-coloring of G that satisfies $\varphi''(u) = \varphi_U(u)$ for each node $u \in U$. \square

Since the described construction preserves both the planarity and the bipartiteness, we obtain the following.

Theorem 8. *The r -FIX problem is NP-complete for any $r \geq 3$, even if the input graph is planar and bipartite.*

This shows that fixing a given coloring remains hard, even if the number of available colors is much bigger than the chromatic number of the input graph.

3.2 Exact algorithm for the FIX problem

In this section we deal with the optimization version of the FIX problem. Note that the brute force algorithm works in time $(\sum_{k=0}^n \binom{n}{k} (r-1)^k) \cdot n^{\mathcal{O}(1)} = r^n \cdot n^{\mathcal{O}(1)}$. We shall obtain a better algorithm by reducing the instance of our problem to an instance of the so-called MAX WEIGHTED PARTITION problem and then solve it, using the algorithm by Björklund, Husfeldt and Koivisto [3]. A *partition* of the set N is a family of sets S_1, \dots, S_r such that $\bigcup_{i=1}^r S_i = N$ and $S_i \cap S_j = \emptyset$ for every $i \neq j$. Notice that we do not require for the sets S_i to be non-empty.

Problem: MAX WEIGHTED PARTITION

Instance: A set N , integer d and functions $f_1, f_2, \dots, f_d: 2^N \rightarrow [-M, M]$ for some integer M .

Question: What is the maximum w , for which there exists a partition S_1, S_2, \dots, S_d , such that $\sum_{i=1}^d f_i(S_i) = w$?

Let G be a graph and let φ be its r -coloring. We shall construct a corresponding instance $\mathcal{J} = (N, d, f_1, \dots, f_d)$ of MAX WEIGHTED PARTITION problem. Set $N = V(G)$ and $d = r$. We define functions f_1, f_2, \dots, f_d as:

$$f_i(S) = \begin{cases} -|S \setminus \varphi^{-1}(i)| & \text{if } S \text{ is independent,} \\ -r \cdot n & \text{otherwise.} \end{cases}$$

In this way every partition of $V(G)$ into r independent set, corresponding to the proper r -coloring φ' , has the total weight $(-\sum_{i=1}^r |\varphi'^{-1}(i) \setminus \varphi^{-1}(i)|)$. It is also easy to notice that any partition into independent sets has greater weight than any partition having at least one non-independent set.

The only thing left is to prove that the weight is maximized for a partition corresponding to a coloring φ' , such that $\text{dist}(\varphi', \varphi)$ is minimum. To see this, notice that $\text{dist}(\varphi', \varphi) = |\{v \in V : \varphi'(v) \neq \varphi(v)\}| = |\bigcup_{i=1}^r \{v \in V : \varphi'(v) = i \wedge \varphi(v) \neq i\}| = \sum_{i=1}^r |\{v \in V : \varphi'(v) = i \wedge \varphi(v) \neq i\}| = \sum_{i=1}^r |\varphi'^{-1}(i) \setminus \varphi^{-1}(i)|$. Moreover, if a weight of a found partition is at most $-r \cdot n$, it contains at least one non-independent set, which means that $r < \chi(G)$ and therefore $\overline{\chi}_\varphi^r(G) = \infty$.

Now we can use the algorithm by Björklund *et al.* to find the optimal solution for \mathcal{J} .

Theorem 9 (Björklund, Husfeldt, Koivisto [3]). MAX WEIGHTED PARTITION problem can be solved in time:

- $2^n d^2 M \cdot n^{\mathcal{O}(1)}$, using exponential space,
- $3^n d^2 M \cdot n^{\mathcal{O}(1)}$, using polynomial space,

where n is the cardinality of the ground set.

We can assume that $r \leq n$, since otherwise we can shift all colors down. Since $d = r$ and $M = n \cdot r$, we obtain the following corollary.

Corollary 10. For any constant r , the optimization version of the FIX problem can be solved in time:

- $2^n \cdot n^{\mathcal{O}(1)}$, using exponential space,
- $3^n \cdot n^{\mathcal{O}(1)}$, using polynomial space,

where n is the number of vertices in the input graph.

It is known, that assuming the ETH, no $2^{o(n)} \cdot n^{\mathcal{O}(1)}$ -algorithm for the r -COLORING problem exists (see e.g. [13]). Thus, by Proposition 6 we immediately obtain the following corollary.

Corollary 11. *For any constant $r \geq 3$, there is no algorithm for the r -FIX problem with running time $2^{o(n)}$ (where n is the number of vertices in the input graph), unless the ETH fails.*

This shows that the algorithms given by Corollary 10 are asymptotically optimal.

4 Parameterized complexity of FIX problem

Since the r -FIX problem is computationally hard, we will turn to parameterized complexity theory, in hope to identify tractable cases.

4.1 Parameterized by the number k of recoloring operations and number r of colors

Clearly for k being a fixed integer, the problem can be easily solved in time $\binom{n}{k} r^k \cdot n^{\mathcal{O}(1)} = n^k (r-1)^k \cdot n^{\mathcal{O}(1)}$. To do it, we have to consider every k -element subset of vertices and check whether recoloring the chosen vertices (in $(r-1)^k$ ways, since we are interested in recoloring *at most* k vertices and thus some colors may remain unchanged) allows us to obtain a proper coloring. Therefore our problem is in XP (when parameterized by $k+r$). In the remainder of this section we show that the problem is in FPT, i.e. we can solve it in time $f(k, r) \cdot n^{\mathcal{O}(1)}$ (note that the degree of the polynomial

function of n does not depend on $k + r$). Consider the following algorithm.

Algorithm 1: $\text{Fix}(r, \mathcal{I} = (G, k, \varphi))$

```

1 if  $\varphi$  is a proper coloring of  $G$  then return YES
2 if  $k = 0$  then return NO
3  $xy \leftarrow$  any edge from  $G^\varphi$ 
4 foreach  $col \in [r] \setminus \{\varphi(x)\}$  do
5    $\varphi_1 \leftarrow \varphi$  with vertex  $x$  recolored to  $col$ 
6   if  $\text{Fix}(r, (G, k - 1, \varphi_1)) = \text{YES}$  then return YES
7 foreach  $col \in [r] \setminus \{\varphi(y)\}$  do
8    $\varphi_1 \leftarrow \varphi$  with vertex  $y$  recolored to  $col$ 
9   if  $\text{Fix}(r, (G, k - 1, \varphi_1)) = \text{YES}$  then return YES
10 return NO

```

Lemma 12. *Let φ be a non-proper r -coloring of G . Then $\mathcal{I} = (G, k, \varphi)$ is a YES-instance of r -FIX if and only if for any edge $xy \in E(G^\varphi)$ there exists an r -coloring (possibly non-proper) φ_1 of G such that:*

1. $\varphi \ominus \varphi_1 = \{x\}$ or $\varphi \ominus \varphi_1 = \{y\}$,
2. $\mathcal{I}' = (G, k - 1, \varphi_1)$ is a YES-instance of r -FIX.

Proof. First assume that $\mathcal{I} = (G, k, \varphi)$ is a YES-instance of r -FIX and let φ' be its witness. Consider an edge xy from G^φ . By the definition of G^φ , we have $\varphi(x) = \varphi(y)$. Since φ' is proper, clearly $\varphi'(x) \neq \varphi'(y)$. Then at least one of the vertices x, y has changed its color. Without loss of generality assume that $\varphi'(x) \neq \varphi(x)$. Let φ_1 be a coloring defined as follows.

$$\varphi_1(u) = \begin{cases} \varphi(u) & \text{if } u \neq x \\ \varphi'(u) & \text{if } u = x. \end{cases}$$

It is clear that it satisfies the conditions given in lemma.

Now consider φ being an r -coloring of G and let xy be an edge from G^φ . Without loss of generality let φ_1 to be some r -coloring of G such that $\varphi \ominus \varphi_1 = \{x\}$ and the instance $\mathcal{I}' = (G, k - 1, \varphi_1)$ is a YES-instance of r -FIX.

Let φ'_1 be a witness of \mathcal{I}' . Notice that $\text{dist}(\varphi'_1, \varphi) \leq \text{dist}(\varphi'_1, \varphi_1) + \text{dist}(\varphi_1, \varphi) \leq k$ and therefore $\mathcal{I} = (G, k, \varphi)$ is a YES-instance of r -FIX with witness φ'_1 . \square

Lemma 13. *The algorithm Fix solves r -FIX problem for any r .*

Proof. Let $\mathcal{I} = (G, k, \varphi)$ be an instance of r -FIX. If φ is a proper labeling of G , then the algorithm returns YES in line 1. Suppose then that φ is not proper. If $k = 0$, the algorithm returns NO in line 2.

Assume that $k > 0$ and the algorithm works properly for all instances with parameter smaller than k . Suppose that $\mathcal{I} = (G, k, \varphi)$ is a YES-instance of r -FIX. Let xy be an edge chosen in line 3. Then, by Lemma 12, there exist an r -coloring φ_1 of G such that $\varphi \ominus \varphi_1 = \{x\}$ or $\varphi \ominus \varphi_1 = \{y\}$ and $\mathcal{I}' = (G, k - 1, \varphi_1)$ is a YES-instance of REC. Without loss of generality assume that $\varphi \ominus \varphi_1 = \{x\}$. Let us consider an iteration of the loop in lines 4–6 color $\varphi_1(x)$. By the inductive assumption, the recursive call in line 6 returns YES and therefore the whole algorithm returns YES.

If \mathcal{I} is a NO-instance of REC, then by the inductive assumption and Lemma 12, for every col the recursive calls in lines 6 and 9 return NO. Therefore the algorithm returns NO in line 10. \square

Let $T(n, k)$ be the computational complexity of the algorithm *Fix*. We can write the following recursive formula:

$$T(n, k) \leq n^{\mathcal{O}(1)} + (r - 1) \cdot T(n, k - 1) + (r - 1) \cdot T(n, k - 1).$$

Solving it, we obtain the following.

Theorem 14. *The algorithm *Fix* solves FIX problem in time $T(n, k) \leq (2(r - 1))^k \cdot n^{\mathcal{O}(1)} = 2^{\mathcal{O}(k \log r)} \cdot n^{\mathcal{O}(1)}$.*

Corollary 15. *The FIX problem is in FPT, when parameterized by $k + r$.*

On the other hand, by Lemma 2 and the fact, that there is no $2^{o(k)} \cdot n^{\mathcal{O}(1)}$ -algorithm for VERTEX COVER, unless the ETH fails [9], we obtain the following lower bound, which shows that our algorithm is almost tight.

Corollary 16. *The FIX cannot be solved in time $2^{o(r+k)} \cdot n^{\mathcal{O}(1)}$, unless the ETH fails.*

4.1.1 Kernelization

Corollary 15 yields that the FIX problem (parameterized by $k + r$) admits a kernel of size $(2(r - 1))^k$ [13]. The next theorem shows, that a polynomial kernel for this problem cannot be obtained (under some standard complexity assumptions).

Theorem 17. *For any $r \geq 3$, the r -FIX parameterized by k of allowed recoloring operations, does not admit a polynomial kernel, unless $NP \subseteq coNP/poly$.*

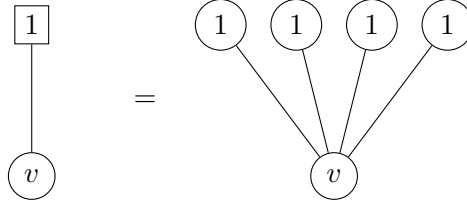


Figure 1: Representation of a 1-fixing gadget for $k = 3$. The square node represent $k + 1$ independent vertices, all colored 1, attached to v .

Proof. Let $t, n > 0$ be two positive integers and let $\mathcal{S} = \{(G_i, U_i, \varphi_{U_i})\}_{i \in [2^t]}$ be a set of 2^t instances of 3-PREEXT such that G_i is bipartite and $|V(G_i)| = n$, for all $i \in [2^t]$. Let $k = t + n + 8$. Our reduction constructs from \mathcal{S} a graph H and a (non-proper) coloring φ of H . This construction satisfies that (H, k, φ) is a YES-instance of r -FIX if and only at least one of the instances in \mathcal{S} is a YES-instance of 3-PREEXT. Since 3-PREEXT on bipartite graphs is NP-complete [24], by Theorem 3 this reduction implies that r -FIX parameterized by the number of allowed recoloring operations does not admit a polynomial kernel, unless $NP \subseteq coNP/poly$.

The construction of H and φ considers three gadgets: the *i-fixing gadget*, the *problem selector gadget*, and the *instance gadget*. The *i-fixing gadget*, for $i \in [r]$, is simply an independent set of $k + 1$ nodes, all colored i in φ , and attached to some vertex v (see Figure 1). If a node v is connected to an *i-fixing gadget*, then in any proper coloring of H obtained from φ with at most k recoloring operations, the color of v is different from i .

For now, suppose that $r = 3$. We will explain how to generalize the construction to the case $r > 3$ at the end of the proof.

The *problem-selector* gadget consists of a complete rooted binary tree of depth t , where the root of the tree is connected to a 1-fixing gadget. The root of the tree is colored 1 in φ , and the rest of the nodes are 3-colored in such a way that the coloring restricted to the nodes in the tree is proper, and the children of a node have different colors in φ (see Figure 2 for an example when $t = 3$). Any proper recoloring obtained from φ after at most $t + 1$ recoloring operations corresponds to a path from the root to a leaf, where each node in the path takes the color of its child also belonging to the path, except for the leaf that is free to pick any color different than its initial one. The leaf picked to be the one that changes its color is called the *selected leaf*. The idea is to attach to each leaf an *instance gadget*, one for each $(G, U, \varphi_U) \in \mathcal{S}$ (note that the number of leaves is 2^t , which is equal to $|\mathcal{S}|$).

The construction of the instance gadget is slightly more complicated. Let (G, U, φ_U) be an instance of 3-PREEXT contained in \mathcal{S} , i.e., $G = (X \cup Y, E)$ is a bipartite graph and U is a subset of $V(G)$, colored according to φ_U . For $j \in [3]$, we define $U_j := \{u \in U : \varphi_U(u) = j\}$. The instance gadget of (G, U, φ_U) is constructed as follows (see Figure 3): pick a copy of G and add ten more nodes called $\{v_1, \dots, v_{10}\}$ with edges $v_1v_2, v_1v_3, v_2v_5, v_2v_6, v_3v_4, v_4v_7$, and v_4v_8 . Moreover, we add all edges v_5u, v_7u for $u \in U_2 \cup U_3$, all edges v_6u, v_8u for $u \in U_1 \cup U_3$, and finally all edges $v_9u, v_{10}u$ for $u \in U_1 \cup U_2$. Then, add 1-fixing gadgets to nodes v_2, v_6, v_8, v_9 and v_{10} . Add 2-fixing gadgets to nodes v_4, v_5, v_7, v_9 and v_{10} . Add 3-fixing gadgets to nodes v_1 and v_3 . Finally, color all the nodes in X , v_1 and v_4 with color 1, all the nodes in Y , v_2 and v_3 with color 2, and $v_5, v_6, v_7, v_8, v_9, v_{10}$ with color 3.

Notice that the instance gadget is properly colored. Suppose that for some reason (this will be forced by the problem-selector gadget) the vertex v_1 must be recolored to a color different than 1. We will show that in that case, we can obtain a proper coloring of the nodes in the instance gadget changing colors of at most $n + 8$ vertices if and only if (G, U, φ_U) is a YES-instance of 3-PREEXT, where $n = |G|$. Indeed, since v_1 is connected with a 3-fixing gadget and it must change its color, it must get color 2. This implies that also v_2 and v_3 must be recolored. Since v_2 is connected to a 1-fixing gadget and v_3 to a 3-fixing gadget, v_2 must change to color 3, and v_3 to color 1. Following the same arguments, v_4 must be recolored 3, and hence v_5 and v_7 must be recolored to 1, and v_6 and v_8 must be recolored to 2. Note that v_9 and v_{10} must stay in color 3 since they are connected with 1-fixing and 2-fixing gadgets. Notice that this situation implies that all nodes in U_i must be recolored i , for $i \in \{1, 2, 3\}$. Therefore, the instance gadget can be properly recolored, if G can be properly colored respecting the pre-coloring of U , i.e. if (G, U, φ_U) is a YES-instance of 3-PREEXT. Note that the number of recoloring operations performed is at most $n + 8$.

Then, we construct H and φ as follows: we start with building the problem-selector gadget. Then we identify each leaf colored 1 with the node v_1 in a copy of the instance gadget built from one of the instances in \mathcal{S} (we have an instance gadget for each instance). For the leafs colored 2 (resp. 3), we make an analogous version of the instance gadget, where the nodes colored 1 change to 2 (resp. 3), the nodes colored 2 change to 3 (resp. 1), the nodes colored 3 change to 1 (resp. 2), and where U_1, U_2, U_3 are changed to U_2, U_3, U_1 (resp. U_3, U_1, U_2). Then we identify each leaf colored 2 or 3 with the node v_1 in a copy of the corresponding version of the instance gadget, built from one of the instances in \mathcal{S} . Note that the size of H is $2^t \cdot (n + 12(k + 1) + 11) + k = \mathcal{O}(|\mathcal{S}|(n + \log |\mathcal{S}|))$, and it can be constructed

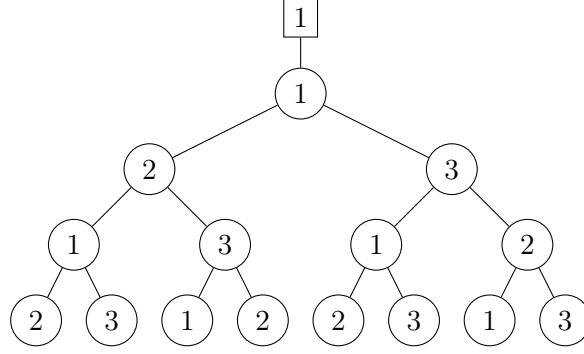


Figure 2: A *problem Selector Gadget*, when $t = 3$. The number in the nodes represent the color of the corresponding node for the coloring φ . The square node represent a 1-fixing gadget.

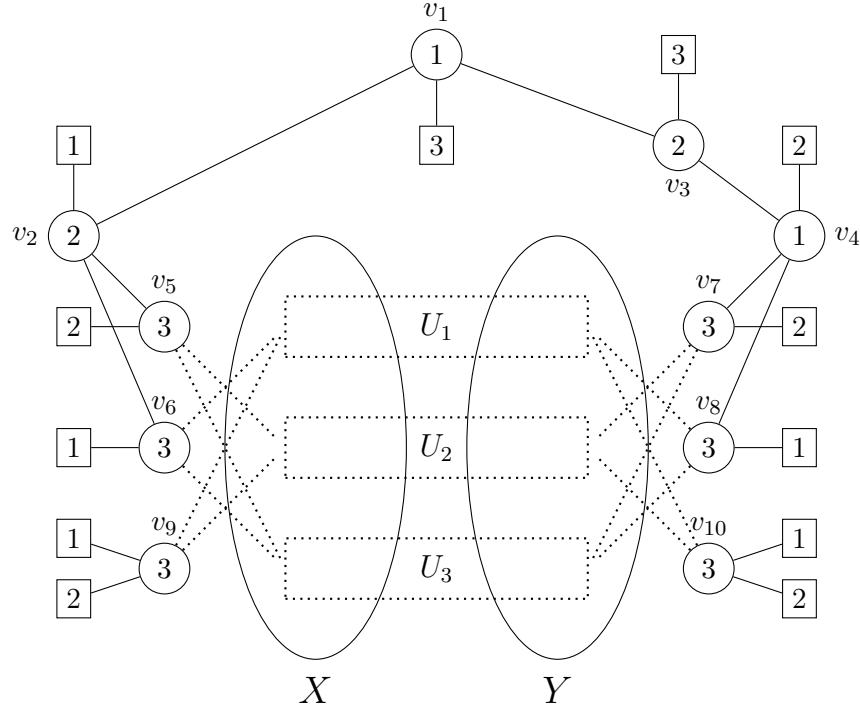


Figure 3: Representation of an *instance gadget* of $G = (X \cup Y, E)$, with precolored set $U = U_1 \cup U_2 \cup U_3$. The numbers in the nodes represent their color in the initial coloring. Square nodes represent fixing gadgets.

in polynomial time in $|S|$ and n .

We claim that $(H, t + n + 8, \varphi)$ is a YES-Instance of 3-FIX if and only if there exists a YES-instance of 3-PREEXT in \mathcal{S} . Indeed, if a recoloring of H from the initial coloring φ forces (by the properties of the problem selector gadget) to pick a path from the root of the binary tree to a leaf, hence recoloring at least t vertices. Recoloring the selected leaf *activates* the corresponding instance gadget, where the recoloring with at most $n + 8$ changes implies that the underlying instance of 3-PREEXT is a YES-instance. In the converse, if there exists a YES-instance of 3-PREEXT in \mathcal{S} , then the coloring of the corresponding instance gadget (requiring to recolor at most $n + 8$ vertices), and later a path from the corresponding leaf to the root (requiring to recolor t vertices), is a witness for the instance $(H, t + n + 8, \varphi)$ of 3-FIX.

To generalize the construction to work for r -FIX for all $r \geq 3$, we can simply add to each vertex in H an i -fixing gadget, for each $i \in [r] \setminus \{1, 2, 3\}$. This forces the nodes in H to pick colors only in $\{1, 2, 3\}$, so everything works as in the case for $r = 3$. \square

Observe that if all instances in \mathcal{S} are bipartite, then also H is bipartite. Thus we obtain the following.

Corollary 18. *For any $r \geq 3$, the r -FIX problem parameterized by k , does not admit a polynomial kernel, unless $NP \subseteq coNP/poly$, even if the input graph is bipartite.*

4.2 Parameterized by the treewidth of the input graph

In this section we consider the optimization version of r -FIX problem for graphs with bounded treewidth. For more information about tree decompositions and treewidth, the reader is referred to Diestel's book [14]. Here we just quickly present basic definitions that we shall use.

Let $G = (V, E)$ be a graph with n vertices. A *tree decomposition* of G is a pair $(\{X_i : i \in I\}, T = (I, F))$, where T is a tree whose every node has associated a subset X_i of vertices of G with the following properties:

1. $\bigcup_{i \in I} X_i = V$,
2. for every $vw \in E$ there exists $i \in I$ such that $\{v, w\} \subseteq X_i$,
3. for every $v \in V$, the set $\{i \in I : v \in X_i\}$ induces a subtree in T .

The *width* of a tree decomposition $(\{X_i : i \in I\}, T)$ is equal to $\max_{i \in I} |X_i| - 1$, while the *treewidth* $\text{tw}(G)$ of a graph G is the minimum width of a tree decomposition of G .

Let T be a rooted tree. This gives us a notion of „children” of nodes of T . A decomposition $(\{X_i : i \in I\}, T = (I, F))$ of $G = (V, E)$ is *nice* if every node $i \in I$ belongs to one of the following types:

1. Leaf: node i is a leaf of T and $|X_i| = 1$,
2. Introduce: node i has exactly one child j and there is a vertex $v \in V$ such that $X_i = X_j \cup \{v\}$,
3. Forget: node i has exactly one child j and there is a vertex $v \in V$ such that $X_j = X_i \cup \{v\}$,
4. Join: node i has exactly two children j_1 and j_2 , and $X_i = X_{j_1} = X_{j_2}$.

Every graph G with n vertices admits a nice tree decomposition with $\mathcal{O}(n)$ nodes and width equal to $\text{tw}(G)$. Moreover, it can be found in linear time if $\text{tw}(G)$ is bounded (for the details see Bodlaender [4] and Kloks [23]).

In the proof of Theorem 19 we shall use a standard technique of dynamic programming on a tree-decomposition. See the survey by Bodlaender and Koster [6] for more examples.

Theorem 19. *For any fixed r , the optimization version of r -FIX problem can be solved in time $r^t \cdot n^{\mathcal{O}(1)}$, where n is the number of vertices of the input graph and t is its treewidth.*

Proof. Consider a graph G and its r -coloring φ . Let $(\{X_i : i \in I\}, T = (I, F))$ be a nice tree decomposition of G with width equal to $\text{tw}(G)$. Let i_0 be the root of T . Moreover, let G_i denote a subgraph of G induced by the set $\bigcup_j X_j$ where j belongs to the subtree of T rooted at i .

For every node i of T we introduce a table K_i , indexed with all possible proper r -colorings of X_i . Let $f : X_i \rightarrow [r]$. If f is a proper r -coloring of $G[X_i]$, then $K_i[f]$ is the minimum number of recolorings needed to obtain from $\varphi|_{G_i}$ a proper r -coloring φ' of G_i , such that $\varphi'|_{X_i} = f$. Clearly, $\overline{\chi}_\varphi^r(G) = \min_f \{K_{i_0}[f]\}$.

We shall show how to construct tables K_i for every type of node. We traverse T in a post-order fashion, so when we consider a node i , we have already considered all its children.

Leaf node. Let i be a leaf node and $X_i = \{v\}$. It is clear that the value of $K_i[f]$ is equal to 0 if $f(v) = \varphi(v)$ or 1 otherwise. Thus the table K_i can be computed in $\mathcal{O}(1)$ time (as r is fixed).

Introduce node. Let i be an introduce node and j be its child node with $X_i = X_j \cup \{v\}$. Observe that from the properties of tree decompositions follows that $v \notin V(G_j)$ (property 3). Moreover, v is not adjacent to any vertex from $V(G_j) \setminus X_j$ (property 2 and 3). It is not hard to observe that $K_i[f]$ is defined as follows:

$$K_i[f] = \begin{cases} K_j[f|_{X_j}] & \text{if } f(v) = \varphi(v) \\ K_j[f|_{X_j}] + 1 & \text{otherwise.} \end{cases}$$

Observe that K_i can be computed in time $\mathcal{O}(r^{t+1})$.

Forget node. Let i be a forget node and j be its child with $X_i \cup \{v\} = X_j$. Clearly $G_i = G_j$. Then $K_i[f]$ is the minimum of $K_j[f']$, where $f'|_{X_i} = f$. Note that there are at most r such colorings f' for each f . Therefore the table K_i can be computed in time $\mathcal{O}(r^{t+2})$.

Join node. Let i be a join node and let j_1 and j_2 be its children. Recall that $X_i = X_{j_1} = X_{j_2}$. From the properties of tree decompositions it follows that $V(G_{j_1}) \cap V(G_{j_2}) = X_i$ (property 3) and no vertex from $V(G_{j_1}) \setminus X_i$ is adjacent to a vertex from $V(G_{j_2}) \setminus X_i$. Therefore we can recolor G_i by recoloring G_{j_1} and G_{j_2} separately and glueing obtain colorings on X_i . Thus $K_i[f] = K_{j_1}[f] + K_{j_2}[f] - (f \ominus \varphi|_{X_i})$. Clearly we can compute K_i in time $\mathcal{O}((t+1)r^{t+1})$.

Observe that for each i , the table K_i has at most r^{t+1} elements. Therefore the space complexity of the algorithm is bounded by $\mathcal{O}(n \cdot r^{t+1})$. Since we can compute each table in time $\mathcal{O}(r^{t+2})$, the total time complexity of the algorithm is $\mathcal{O}(n \cdot r^{t+2}) = r^t \cdot n^{\mathcal{O}(1)}$, which finishes the proof of Theorem 19. \square

Lokshtanov *et al.* [25] have proven that, assuming the SETH, the simple $r^t \cdot n^{\mathcal{O}(1)}$ -time algorithm for the r -coloring of an n -vertex graph of treewidth at most t is asymptotically optimal.

Theorem 20 (Lokshtanov *et al.* [25]). *For any $r \geq 3$, the r -COLORING problem cannot be solved in time $(r - \epsilon)^t \cdot n^{\mathcal{O}(1)}$, where n is the number of vertices of the input graph and t is its treewidth, unless the SETH fails.*

By Proposition 6, we observe that the algorithm given by Theorem 19 is asymptotically optimal as well.

Corollary 21. *For any fixed $r \geq 3$ and any $\epsilon > 0$, there is no algorithm for the r -FIX problem with running time $(r - \epsilon)^t \cdot n^{\mathcal{O}(1)}$, where n is the number of vertices of the input graph and t is its treewidth, unless the SETH fails.*

5 Fixing number

Recall that for a graph G and its r -coloring φ , by $\overline{\chi}_\varphi^r(G)$ we denote the minimum number vertices that have to be recolored to obtain some proper r -coloring of G .

An r -fixing number of a graph G (denoted by $\overline{\chi}^r(G)$) is a maximum value of $\overline{\chi}_\varphi^r(G)$ over all colorings $\varphi: V(G) \rightarrow \{1, \dots, r\}$.

Definition 1. By $\overline{\chi}(G)$ we denote the fixing number of a graph G , defined as a maximum value of $\overline{\chi}^r(G)$ over all $r \geq \chi(G)$.

Lemma 22. Let φ be some r -coloring of G and φ' be an $(r+1)$ -coloring of G such that $\varphi^{-1}(i) = \varphi'^{-1}(i)$ for $i \in \{1, \dots, r-1\}$. Then $\overline{\chi}_{\varphi'}^{r+1}(G) \leq \overline{\chi}_\varphi^r(G)$.

Proof. Recoloring the vertices in the same way as with φ makes φ' proper. \square

Lemma 23. For all graphs G and $r \geq \chi(G)$ holds $\overline{\chi}^{r+1}(G) \leq \overline{\chi}^r(G)$.

Proof. Let φ' be an $(r+1)$ -coloring of G such that $\overline{\chi}_{\varphi'}^{r+1}(G) = \overline{\chi}^{r+1}(G)$. Let φ be an r -coloring of G obtained from φ' by identifying colors r and $r+1$. By Lemma 22 we obtain the following. $\overline{\chi}^{r+1}(G) = \overline{\chi}_{\varphi'}^{r+1}(G) \leq \overline{\chi}_\varphi^r(G) \leq \overline{\chi}^r(G)$. \square

Corollary 24. For all graphs G holds $\overline{\chi}(G) = \overline{\chi}^{\chi(G)}(G)$.

Let G be a bipartite graph with bipartition classes X, Y and let φ be its 2-coloring. Recall from Observation 5 that

$$\overline{\chi}_\varphi^r(G) = \sum_{\substack{C: \text{ connected} \\ \text{component of } G}} \min\{ |(X \ominus \varphi^{-1}(1)) \cap V(C)|, |(X \ominus \varphi^{-1}(2)) \cap V(C)| \}.$$

Note that if $|(X \ominus \varphi^{-1}(1)) \cap V(C)| \geq |C|/2$, then $|(X \ominus \varphi^{-1}(2)) \cap V(C)| \leq |C|/2$ for any connected component C of G . Thus we can easily obtain the following corollary.

Corollary 25. $\overline{\chi}(G) \leq \lfloor n/2 \rfloor$ for every bipartite graph G on n vertices.

This result can be generalized for non-bipartite graphs.

Theorem 26. For all G holds $\overline{\chi}(G) \leq \left\lfloor n \cdot \frac{\chi(G)-1}{\chi(G)} \right\rfloor$.

Proof. For a graph $G = (V, E)$ set $r = \chi(G)$ and let φ an r -coloring of G such that $\overline{\chi}_\varphi^r(G) = \overline{\chi}(G)$. Consider some proper r -coloring φ' of G . Let $A_i = \varphi^{-1}(i)$ and $A'_i = \varphi'^{-1}(i)$ for all $i \in [r]$. By $B_{i,j}$ we denote $A_i \cap A'_j$. Clearly $\bigcup_{i,j} B_{i,j} = V$. Note that for any permutation σ of $[r]$, we can obtain a proper r -coloring of G from φ by recoloring all vertices but $C_\sigma = \bigcup_{i \in [r]} B_{i,\sigma(i)}$ (this proper coloring will be equivalent to φ' up to the permutation of colors).

Suppose that $|C_\sigma| < \frac{n}{r}$ for all σ . On one hand we have $|\bigcup_\sigma C_\sigma| \leq \sum_\sigma |C_\sigma| < r! \cdot \frac{n}{r}$. On the other hand, we have:

$$\begin{aligned} |\bigcup_\sigma C_\sigma| &= |\bigcup_\sigma \bigcup_i B_{i,\sigma(i)}| = |\bigcup_i \bigcup_\sigma B_{i,\sigma(i)}| = |\bigcup_i \bigcup_j \bigcup_{\substack{\sigma \text{ s.t.} \\ \sigma(i)=j}} B_{i,j}| \\ &= (r-1)! |\bigcup_i \bigcup_j B_{i,j}| = (r-1)!n. \end{aligned}$$

This is a contradiction, so there exists σ with $|C_\sigma| \geq \frac{n}{r}$ and therefore we can obtain a proper r -coloring of G by recoloring at most $n \cdot \frac{r-1}{r}$ vertices. Since $\overline{\chi}(G)$ is an integer, we obtain our claim. \square

To see that this bound is attainable, consider a graph $G(m, r)$ on $n = m \cdot r$ vertices, consisting of m disjoint copies of K_r . Clearly $\chi(G(m, r)) = r$. Let φ be an r -coloring of $G(m, r)$ such that $\varphi(v) = 1$ for every vertex v . Clearly we have to recolor every vertex but one from every copy of K_r , which gives us $\overline{\chi}(G(m, r)) \geq \overline{\chi}_\varphi^r(G(m, r)) = m(r-1) = n \cdot \frac{r-1}{r}$.

However, there are graphs G for which the value of $\overline{\chi}(G)$ is significantly smaller. For example, consider an odd cycle C_n for $n \geq 9$. Clearly $\chi(C_n) = 3$. Let φ be any coloring of C_n with $r \geq 3$ colors. Arbitrarily choose vertex v and remove it from C_n , obtaining a path P_{n-1} . Since $\chi(P_{n-1}) = 2$, then by Theorem 26 we can obtain a proper coloring of P_{n-1} by recoloring at most $\lfloor (n-1)/2 \rfloor$ vertices. Then we can restore vertex v and, if necessary, recolor it to an available color (there is always at least one). In this way we performed at most $1 + \lfloor (n-1)/2 \rfloor$ recoloring operations, which is roughly $\frac{n}{2}$ compared to $\frac{2n}{3}$ given by Theorem 26.

We believe that it would be interesting to find some reasonable graph classes, for which the bound from Theorem 26 is not tight.

Another direction of research is to find better lower bounds for the fixing number. In particular, we give the following conjecture.

Conjecture 1. *For all G , it holds that $\overline{\chi}(G) \geq n/\chi(G)$.*

Acknowledgement. The authors are sincerely grateful to Dieter Kratsch for valuable discussion on the topic.

References

- [1] Ausiello G., Escoffier B., Monnot J., Paschos V.Th.: Reoptimization of minimum and maximum traveling salesmans tours. *J. of Discrete Algorithms* 7 (2009), pp. 453–463
- [2] Bilò D., Böckenhauer H.-J., Komm D., Královic R., Mömke T., Seibert S., Zych A.: Reoptimization of the Shortest Common Superstring Problem. *Algorithmica* 61 (2011), pp. 227–251
- [3] Björklund A., Husfeldt T., Koivisto M.: Set partitioning via inclusion-exclusion. *SIAM Journal on Computing* 39 (2009), pp. 546–563
- [4] Bodlaender H.: A linear time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing* 25 (1996), pp. 1305–1317
- [5] Bodlaender H., Jansen B., Kratsch S.: Cross-composition: A new technique for kernelization lower bounds. *Proc. of STACS 2011, LIPIcs* 9, pp. 165–176.
- [6] Bodlaender H., Koster A.: Combinatorial Optimization on Graphs of Bounded Treewidth. *The Computer Journal* (2007)
- [7] Bonamy M., Bousquet N.: Recoloring bounded treewidth graphs. *Electronic Notes in Discrete Mathematics* 44 (2013), pp. 257–262
- [8] Bonsma P., Cereceda L.: Finding paths between graph colourings: PSPACE-completeness and superpolynomial distances. *Proc. of MFCS 2007, LNCS 4708* (2007), pp. 738–749
- [9] Cai L., Juedes W.: On the existence of subexponential parameterized algorithms. *J. Comput. Syst. Sci.* 67 (2003), pp. 789–807
- [10] Cereceda L., Heuvel, J. van den, Johnson M.: Mixing 3-colourings in bipartite graphs. *Proc. of WG 2007, LNCS 4769* (2007), pp. 166–177
- [11] Cereceda L., Heuvel, J. van den, Johnson M.: Connectedness of the graph of vertex colourings. *Discrete Mathematics* 308 (2008). pp. 166–177

- [12] Cereceda L., Heuvel, J. van den, Johnson M.: Finding paths between 3-colorings. *Journal of Graph Theory* 67 (2011). pp. 69-82
- [13] Cygan M., Fomin F., Kowalik Ł., Lokshtanov D., Marx D., Pilipczuk M., Pilipczuk M., Saurabh S.: Parameterized Algorithms. Springer (2015)
- [14] Diestel R.: Graph Theory (3rd ed.). *Graduate Texts in Mathematics*, Springer (2005)
- [15] Downey R.G., Fellows M.R.: Parameterized Complexity. *Monographs in Computer Science*, Springer (1999)
- [16] Felsner S., Huemer C., Saumell M.: Recoloring directed graphs. *Proc. of XIII Encuentros de Geometría Computacional* (2009), pp. 91–97
- [17] Garey M., Johnson D.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman & Co. (1979).
- [18] Impagliazzo R., Paturi R.: On the complexity of k -SAT. *Journal of Computer and System Sciences* 62 (2001), pp. 367 – 375
- [19] R. Impagliazzo, R. Paturi, and F. Zane.: Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.* 63 (2001), pp. 512–530
- [20] Ito T., Demaine E., Harvey N.J.A., Papadimitriou C.H., Sideri M., Uehara R., Uno Y.: On the complexity of reconfiguration problems. *Theoretical Computer Science* 412 (2011), pp. 1054–1065
- [21] Jerrum M.: A very simple algorithm for estimating the number of k -colorings of a low-degree graph. *Random Structures & Algorithms* 7 (1995), pp. 157–165
- [22] Junosza-Szaniawski K., Liedloff M., Rzażewski P.: Fixing improper colorings of graphs. *Proc. of SOFSEM 2015, LNCS 8939* (2015), pp. 266–276
- [23] Kloks T.: Treewidth. Computations and Approximations. *LNCS 842*, Springer (1994)
- [24] Kratochvíl, J.: Precoloring extension with fixed color bound. *Acta Mathematica Universitatis Comenianae. New Series*, Comenius University Press (1993)

- [25] Lokshstanov D., Marx D., Saurabh S.: Known Algorithms on Graphs of Bounded Treewidth are Probably Optimal. *Proc. of SODA 2011, SIAM* (2011), pp. 760–776
- [26] Lokshstanov D., Marx D., Saurabh S.: Lower bounds based on the exponential time hypothesis. *Bulletin of the EATCS 105* (2011), pp. 41–72
- [27] Shachnai H., Tamir G., Tamir T.: A Theory and Algorithms for Combinatorial Reoptimization. *Proc. of LATIN 2012, LNCS 7256* (2012), pp. 618–630
- [28] Zych A., Bilò D.: New Reoptimization Techniques applied to Steiner Tree Problem. *Electronic Notes in Discrete Mathematics 37* (2–1), pp. 387–392